Data Description:

Parameter	Description	Content type
age	Age in years	integer
gender	Male or Female	integer (1 or 2)
bmi	Body mass index	float
no_of_children	Number of children	integer
smoker	Whether smoker or not	integer (0 or 1)
region	Which US region - NW, NE, SW, SE	integer (1,2,3 or 4 respectively)
charges	Annual Insurance charges in USD	float

Objectives

In this project, you will:

- Load the data as a `pandas` dataframe
- Clean the data, taking care of the blank entries
- Run exploratory data analysis (EDA) and identify the attributes that most affect the `charges`
- Develop single variable and multi variable Linear Regression models for predicting the `charges`
- Use Ridge regression to refine the performance of Linear regression models.

Setup

For this lab, we will be using the following libraries:

- `pandas` for managing the data.
- `numpy` for mathematical operations.
- `sklearn` for machine learning and machine-learning-pipeline related functions.
- `seaborn` for visualizing the data.
- `matplotlib` for additional plotting tools.

Importing Required Libraries

We recommend you import all required libraries in one place (here):

In [1]:
```
import pandas as pd
import numpy as np
```

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LinearRegression,Ridge
from sklearn.preprocessing import StandardScaler, PolynomialFeatures
from sklearn.metrics import mean_squared_error,r2_score
from sklearn.model_selection import cross_val_score, train_test_split
```

Task 1 : Import the dataset

Import the dataset into a `pandas` dataframe. Note that there are currently no headers in the CSV file.

Print the first 10 rows of the dataframe to confirm successful loading.

In [3]:
```
df=pd.read_csv('https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
df.head()
```

Out[3]:

	0	1	2	3	4	5	6
0	19	1	27.900	0	1	3	16884.92400
1	18	2	33.770	1	0	4	1725.55230
2	28	2	33.000	3	0	4	4449.46200
3	33	2	22.705	0	0	1	21984.47061
4	32	2	28.880	0	0	1	3866.85520

Add the headers to the dataframe.

In [4]:
```
headers = ["age", "gender", "bmi", "no_of_children", "smoker", "region", "charges"]
df.columns=headers
```

Now, replace the '?' entries with 'NaN' values.

In [7]:
```
df=df.replace('?',np.nan)
```

Task 2 : Data Wrangling

Use `dataframe.info()` to identify the columns that have some 'Null' (or NaN) information.

In [9]:
```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2772 entries, 0 to 2771
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   age             2768 non-null   object 
 1   gender          2772 non-null   int64  
 2   bmi             2772 non-null   float64
 3   no_of_children  2772 non-null   int64  
 4   smoker          2765 non-null   object 
 5   region          2772 non-null   int64  
 6   charges         2772 non-null   float64
dtypes: float64(2), int64(3), object(2)
memory usage: 151.7+ KB
None
```

Handle missing data:

- For continuous attributes (e.g., age), replace missing values with the mean.
- For categorical attributes (e.g., smoker), replace missing values with the most frequent value.
- Update the data types of the respective columns.
- Verify the update using `df.info()`.

In [12]:
```
df.isnull().sum()
```

Out[12]:
```
age               4
gender            0
bmi               0
no_of_children    0
smoker            7
region            0
charges           0
dtype: int64
```

In [25]:
```
average_value=df['age'].astype(float).mean()
df['age'].replace(np.nan,average_value,inplace=True)
```

In [27]:
```
freq_value=df['smoker'].value_counts().idxmax()
df['smoker'].replace(np.nan,freq_value,inplace=True)
```

In [28]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2772 entries, 0 to 2771
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   age             2772 non-null   object 
 1   gender          2772 non-null   int64  
 2   bmi             2772 non-null   float64
 3   no_of_children  2772 non-null   int64  
 4   smoker          2772 non-null   object 
 5   region          2772 non-null   int64  
 6   charges         2772 non-null   float64
dtypes: float64(2), int64(3), object(2)
memory usage: 151.7+ KB
```

Also note, that the `charges` column has values which are more than 2 decimal places long. Update the `charges` column such that all values are rounded to nearest 2 decimal places. Verify conversion by printing the first 5 values of the updated dataframe.

In [34]:
```
df['charges']=df['charges'].round(2)
df.head(5)
```

Out[34]:

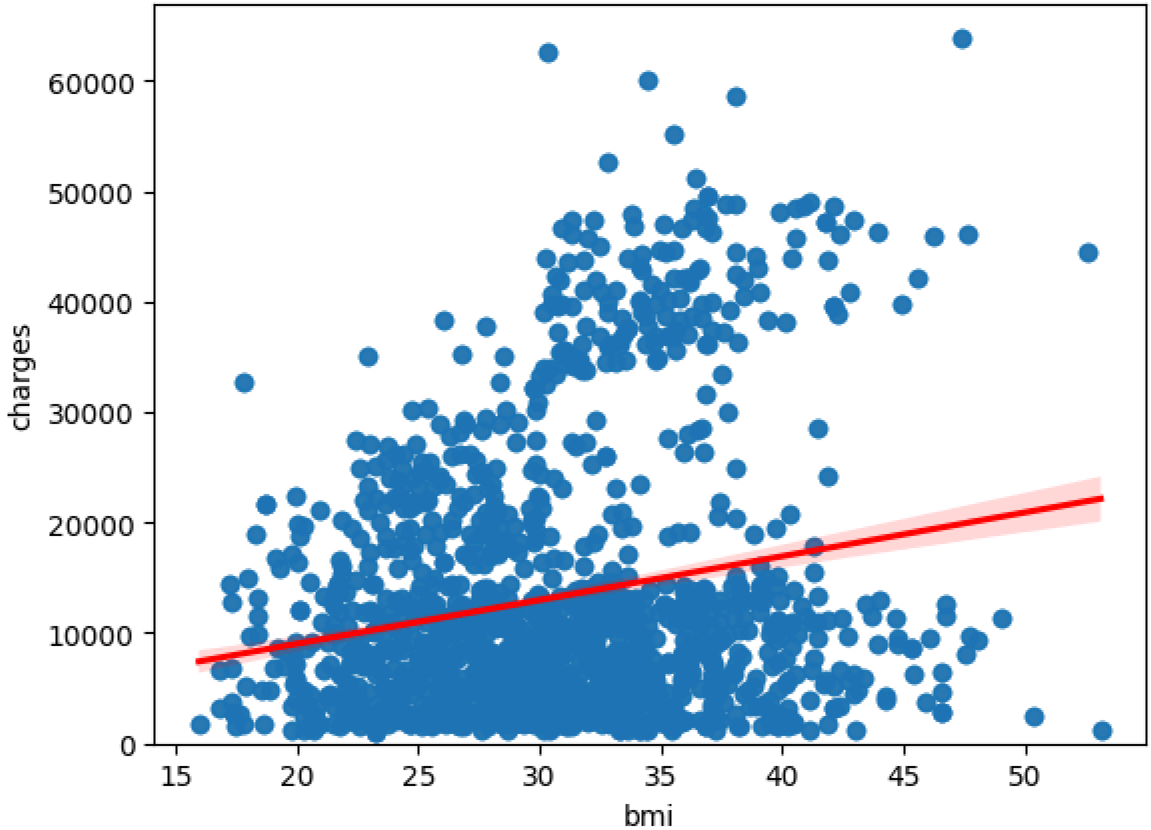
	age	gender	bmi	no_of_children	smoker	region	charges
0	19	1	27.900	0	1	3	16884.92
1	18	2	33.770	1	0	4	1725.55
2	28	2	33.000	3	0	4	4449.46
3	33	2	22.705	0	0	1	21984.47
4	32	2	28.880	0	0	1	3866.86

Task 3 : Exploratory Data Analysis (EDA)

Implement the regression plot for `charges` with respect to `bmi`.

In [39]:
```
sns.regplot(x="bmi", y="charges", data=df, line_kws={"color": "red"})
plt.ylim(0,)
```

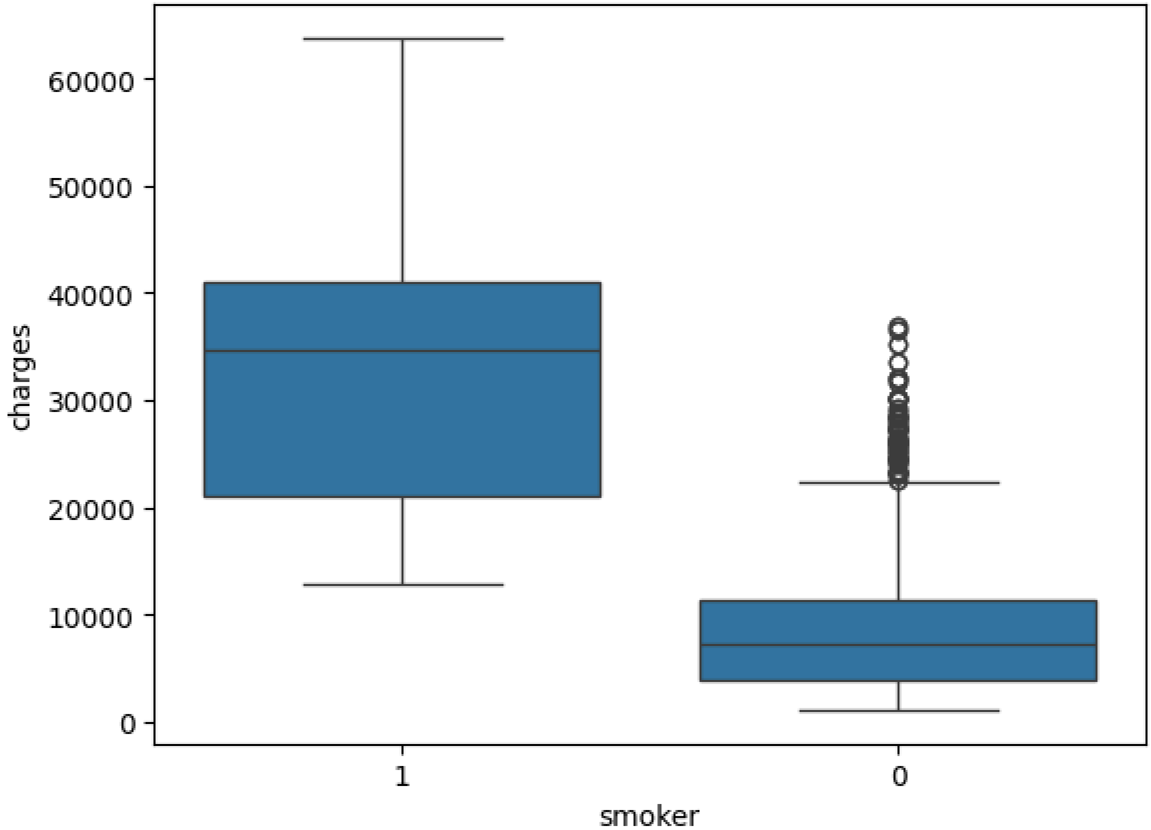
Out[39]: (0.0, 66902.85800000001)



Implement the box plot for `charges` with respect to `smoker`.

In [42]:
```
sns.boxplot(x="smoker", y="charges", data=df)
```

Out[42]:
```
<Axes: xlabel='smoker', ylabel='charges'>
```



Print the correlation matrix for the dataset.

In [43]:
```
df.corr()
```

Out[43]:

	age	gender	bmi	no_of_children	smoker	region	charg
age	1.000000	-0.026041	0.113045	0.037585	-0.023285	-0.007175	0.2986
gender	-0.026041	1.000000	0.042924	0.016020	0.082326	0.022213	0.0628
bmi	0.113045	0.042924	1.000000	-0.001492	0.011489	0.271119	0.1998
no_of_children	0.037585	0.016020	-0.001492	1.000000	0.006362	-0.025717	0.0664
smoker	-0.023285	0.082326	0.011489	0.006362	1.000000	0.054077	0.7887
region	-0.007175	0.022213	0.271119	-0.025717	0.054077	1.000000	0.0540
charges	0.298622	0.062837	0.199846	0.066442	0.788783	0.054058	1.0000

Task 4 : Model Development

Fit a linear regression model that may be used to predict the `charges` value, just by using the `smoker` attribute of the dataset. Print the R^2 score of this model.

In [46]:
```
lr=LinearRegression()
x=df[['smoker']]
y=df['charges']
lr.fit(x,y)
print(lr.score(x,y))
```

```
0.6221791733924185
```

Fit a linear regression model that may be used to predict the `charges` value, just by using all other attributes of the dataset. Print the R^2 score of this model. You should see an improvement in the performance.

In [47]:
```
z=df.drop('charges',axis=1)
lr.fit(z,y)
print(lr.score(z,y))
```

```
0.7504063772187107
```

Create a training pipeline that uses `StandardScaler()`, `PolynomialFeatures()` and `LinearRegression()` to create a model that can predict the `charges` value using all the other attributes of the dataset. There should be even further improvement in the performance.

In [50]:
```
Input=[('scale',StandardScaler()),('feature',PolynomialFeatures(include_bias=False)
pipe=Pipeline(Input)
z=z.astype(float)
pipe.fit(z,y)
ypipe=pipe.predict(z)
print(r2_score(y,ypipe))
```

```
0.845253412435446
```

Task 5 : Model Refinement

Split the data into training and testing subsets, assuming that 20% of the data will be reserved for testing.

In [52]:
```
x_train, x_test, y_train, y_test = train_test_split(z, y, test_size=0.2, random_sta
```

Initialize a Ridge regressor that used hyperparameter $\alpha = 0.1$. Fit the model using training data data subset. Print the R^2 score for the testing data.

In [54]:
```
ridge=Ridge(alpha=0.1)
ridge.fit(x_train,y_train)
yhat=ridge.predict(x_test)
print(r2_score(y_test,yhat))
```

```
0.7395711241633176
```

Apply polynomial transformation to the training parameters with degree=2. Use this transformed feature set to fit the same regression model, as above, using the training subset. Print the R^2 score for the testing subset.

In [56]:
```
pr=PolynomialFeatures(degree=2)
x_train_pr = pr.fit_transform(x_train)
x_test_pr = pr.fit_transform(x_test)
ridge.fit(x_train_pr, y_train)
y_hat = ridge.predict(x_test_pr)
print(r2_score(y_test,y_hat))
```

```
0.8339381387536698
```